Applying Data Compression Techniques on Systolic Neural Network Accelerator

Navid Mirnouri

Amirkabir University of Technology, Hafez St., Tehran, Iran
n72m@aut.ac.ir

Technical Report
September 2016

Abstract. New directions in computing and algorithms has lead to some new applications that have tolerance to imprecision. Although, These applications are creating large volumes of data which exceeds the capability of today's computing systems. Therefore, researchers are trying to find new techniques to alleviate this crisis. Approximate Computing is one promising technique that uses a trade off between precision and efficiency of computing. Acceleration is another solution that uses specialized logics in order to do computations in a way that is more power efficient. Another technique is Data compression which is used in memory systems in order to save capacity and bandwidth.

Keywords: Approximate Computing, Neural Acceleration, Data Compression , SNNAP

1 Introduction

Emerging applications in mining, learning and big data have created a large workload for computing systems. While technology improvement is diminishing, designers are trying to find alternative solutions for this crisis[1]. Approximate Computing is a promising solution, which utilize resilience of application in order to gain efficiency in all layers of computing stack[2]. These techniques can be applied in each layer of computing: application, architecture and circuit design. Figure1 illustrate this hierarchical architecture. Specialized logic including accelerators and programmable logic is another solution[2]. In some previous works, neural networks were used as an accelerator in order to be an alternative for certain part of code[3]. SNNAP is a neural accelerator which uses FPGA as an accelerator for implementation of Neural Processing Units (NPUs)[1]. SNNAP has been implemented on Zynque board which is a Field Programmable logic (FPGA). SNNAP uses ACP port for communication between NPUs and the processor. We believe that by applying some techniques including data compression, effective memory bandwidth could be increased. Linearly Compressed Page(LCP)[4] is one solution, which uses two compression techniques: Based

Delta Immediate Compression[5] and Frequent Pattern Compression[6]. In future section we give a short review on approximate computing in different layer. Then we investigate SNNAP, after that we study Data compression techniques including: Based Data Immediate, Frequent Pattern and Linearly Compressed Page.

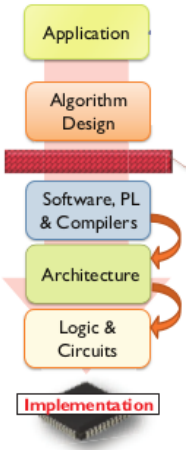

Fig. 1. Layers of approximate computing[2].

2 Approximate Computing

In today computing systems, energy efficiency is a key concern. Some application like mining, image processing and learning have tolerance to imprecision. Designing a platform for approximate computing is a trade off between energy-efficiency and precision[7]. At circuit design level there some approaches like time scaling or voltage scaling have been proposed[8]. Different approaches at architectural level could be categorized in three groups[7]: application specific accelerators, programmable processors and approximate memories. In [9] a programmable processor has been proposed which uses ISA extension to use some ISA for specifying the accuracy of processing. Approximation in software level can reduce the run-time of application, it could be achieved via different approaches[9]. Specifying the part of the code to be executed approximately is an important challenge[9].

3 Neural Networks

Neural network accelerators has been proposed in previous works[3][11][14]. The key idea is to specify part of the code that have the tolerance to imprecision and

transform them to Neural Networks. Implementation of neural networks can be both in hardware[3][12][13] or software[11][14].

3.1 Systolic Neural Network accelerator in Programmable logics[3]

SNNAP is a neural network accelerator which uses programmable system on chips (PSocs) to implement Neural Processing units. PsoCs are heterogeneous architectures which have both hard processor and programmable logics on the same die[3]. Integration of the CPU and programmable logic can be beneficial for interconnections' bandwidth comparing to traditional FPGAs.

Some important challenges in SNNAP should be mentioned before going through more details.
1. Neural processing unit should be implemented on FPGA in a way that uses resources to minimize energy consumption.
2. Communication between NPU and processor should be low latency in order to be efficient for those applications which have smaller part of approximate code. Processing batches of requests is one solution.
3. In order for the system to be more energy efficient, Processor and NPU should be able to work independently. For example, when NPU is working CPU can be hibernated with special commands.
4. After transformation,different applications have different Nns. This variation in neural network topology should be handled appropriately without much of hardware effort and reprogramming the FPGA.

SNNAP invokes MLP which is a multi-layer directed graph[3]. For implementation of MLP, SNNAP uses Digital Signal Processing(DSP) slices of FPGA.

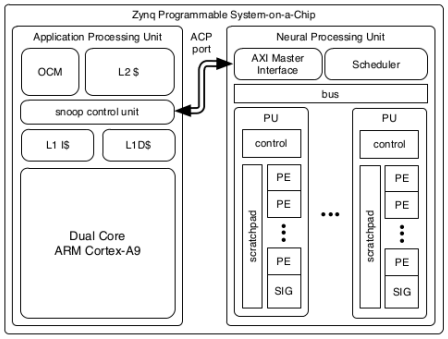

Fig. 2. Architectrual Design of SNNAP[3].

4 Data Compression

Data compression is an approach to reduce capacity and bandwidth requirements by recognizing redundancy of data in applications [10]. These techniques can be applied in order to improve cache performance, memory bandwidth or interconnections. In this section we are going to study some of these techniques.

Base-delta-immediate is a compression technique for increasing on-chip cache's capacity. Data in cache lines are not wide in range. The key idea in BDI is to use low dynamic range of data in cache line and represent data with a base value and an array of deltas[5]. Frequent Pattern Compression (FPC) realize that large volumes of records in memory are occurring frequently. This technique tries to compress these record with fewer bits [4]. Linearly Compressed Page (LCP) is a technique which uses two previous algorithm to gain more efficiency than prior works [4]. The key idea in LCP is using a fixed size for each compressed block in order to eliminate complexity of calculating the location of each block in memory. Prior works, used variable-sized of cache line, and this approach can cause a lot of long-latency and complex main memory address calculations[4].

5 Future Works

In this report some hot topics in computer architecture has been studied briefly. Approximate Computing is a technique that tries to benefit from imprecision tolerance in some applications like mining, learning in order to do computing in a more efficient way. This technique can be applied in different layers of abstraction from application to circuit design.

Neural Acceleration is another direction that tries to invoke Neural Networks in either software or hardware form to improve the efficiency of computing. It selects certain part of the code and transform them to Neural Network. Implementation of Neural networks can be in hardware [3][12][13]or software [11][14].

We believe that SNNAP have some potentials for future works. SNNAP is using multi-layered perceptron (MLP), we believe that it can be implemented more efficiently if we use certain classes of Neural networks for different category of applications. There has been a study on implementation of PARSEC and accelerating it with Neural Networks[11]. BenchNN uses different kinds of Neural Networks for each benchmark in PARSEC. For example for canneal benchmark, BenchNN uses Hopfield Neural Network (HNN) instead of MLP[11]. We believe that future works can address the possibility of customized NPU implementation in SNNAP.

Another direction is to study data compression techniques that can be applied to SNNAP for improving memory bandwidth. Some of them have been mentioned briefly in this paper, but there are more research opportunities to find a practical data compression that can be applied to SNNAP.

References

1. Moreau, T., Wyse, M., Nelson, J., Sampson, A., Esmaeilzadeh, H., Ceze, L., Oskin, M. (2015, February). SNNAP: Approximate computing on programmable SoCs via neural acceleration. In 2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA) (pp. 603-614). IEEE.

2. Venkataramani, S., Chakradhar, S. T., Roy, K., Raghunathan, A. (2015, June). Approximate computing and the quest for computing efficiency. In Proceedings of the 52nd Annual Design Automation Conference (p. 120). ACM.

3. Esmaeilzadeh, H., Sampson, A., Ceze, L., Burger, D. (2012, December). Neural acceleration for general-purpose approximate programs. In Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture (pp. 449-460). IEEE Computer Society.

4. Pekhimenko, G., Mowry, T. C., Mutlu, O. (2012, September). Linearly compressed pages: A main memory compression framework with low complexity and low latency. In Proceedings of the 21st international conference on Parallel architectures and compilation techniques (pp. 489-490). ACM.

5. Pekhimenko, G., Seshadri, V., Mutlu, O., Mowry, T. C., Gibbons, P. B., Kozuch, M. A. (2012). Base-delta-immediate compression: A practical data compression mechanism for on-chip caches. PACT-21.

6. Alameldeen, A. R., Wood, D. A. (2004). Frequent pattern compression: A significance-based compression scheme for L2 caches. Dept. Comp. Scie., Univ. Wisconsin-Madison, Tech. Rep, 1500.

7. Han, J., Orshansky, M. (2013, May). Approximate computing: An emerging paradigm for energy-efficient design. In 2013 18th IEEE European Test Symposium (ETS) (pp. 1-6). IEEE.

8. Chippa, V. K., Mohapatra, D., Roy, K., Chakradhar, S. T., Raghunathan, A. (2014). Scalable effort hardware design. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 22(9), 2004-2016.

9. Venkataramani, S., Chippa, V. K., Chakradhar, S. T., Roy, K., Raghunathan, A. (2013, December). Quality programmable vector processors for approximate computing. In Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture (pp. 1-12). ACM.

10. Yazdanbakhsh, A., Pekhimenko, G., Thwaites, B., Esmaeilzadeh, H., Mutlu, O., Mowry, T. C. (2016). RFVP: Rollback-free value prediction with safe-to-approximate loads. ACM Transactions on Architecture and Code Optimization (TACO), 12(4), 62.

11. Chen, T., Chen, Y., Duranton, M., Guo, Q., Hashmi, A., Lipasti, M., ... Temam, O. (2012, November). BenchNN: On the broad potential application scope of hardware neural network accelerators. In Workload Characterization (IISWC),

2012 IEEE International Symposium on (pp. 36-45). IEEE.

12. Merolla, P., Arthur, J., Akopyan, F., Imam, N., Manohar, R., Modha, D. S. (2011, September). A digital neurosynaptic core using embedded crossbar memory with 45pJ per spike in 45nm. In 2011 IEEE custom integrated circuits conference (CICC) (pp. 1-4). IEEE.
13. Seo, J. S., Brezzo, B., Liu, Y., Parker, B. D., Esser, S. K., Montoye, R. K., ... Friedman, D. J. (2011, September). A 45nm CMOS neuromorphic chip with a scalable architecture for learning in networks of spiking neurons. In 2011 IEEE Custom Integrated Circuits Conference (CICC) (pp. 1-4). IEEE.
14. Grigorian, B., Reinman, G. (2015). Accelerating divergent applications on simd architectures using neural networks. ACM Transactions on Architecture and Code Optimization (TACO), 12(1), 2.